KÄYTETTÄVYYSTESTAUS OSANA KETTERÄÄ OHJELMISTOKEHITYSTÄ

Case Polar Electro Oy

Heli Puikkonen

Opinnäytetyö
Marraskuu 2011

Teietojenkäsittelyn koulutusohjelma
Luonnontieteiden ala

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖN
KUVAILULEHTI

Tekijä(t) PUIKKONEN, Heli	Julkaisun laji Opinnäytetyö	Päivämäärä 21.11.2011
	Sivumäärä 35	Julkaisun kieli suomi
	Luottamuksellisuus ( ) saakka	Verkkojulkaisulupa myönnetty (X)

Työn nimi
Käytettävyystestaus osana ketterää ohjelmistokehitystä

Koulutusohjelma

Tietojenkäsittely

Työn ohjaaja(t)
Kiviaho, Niko

Toimeksiantaja(t)
Polar Electro Oy

Tiivistelmä

Käytettävyystestaus osana ketterää ohjelmistokehitystä on tilattu opinnäytetyö. Työ on toteutettu Polar Electo Oy:lle, ja sen ensisijaisena tarkoituksena oli kehittää ja tuoda esiin menetelmä jokapäiväisen käytettävyystestauksen suorittamiseen ketterässä ohjelmistokehitysprojektissa, juuri kyseisen yrityksen tarpeita varten. Menetelmän tarkoitus on helpottaa käytettävyyden parantamista pääasiallisesti lopullista tuotetta ajatellen ennen sen käyttöönottoa.

Yrityksen lähtötila käytettävyyden mittaamiseen ja sen testaamiseen kartoitettiin kyselytutkimuksella, johon vastanneet koostuivat valikoidusti asiantuntijoista yrityksen sisällä, useista eri tiimeistä ja osastoista. Kyselytutkimuksen tärkeimpänä esiin tulleena asiana selvisi, ettei yrityksellä ole tällä hetkellä mitään tiettyä, yhtenäistä mallia ohjelmistotuotannon projekteissa. Teoriaosuutta koottaessa tuli esille, kuinka selvästi pienilläkin resursseilla voidaan vaikuttaa suuresti käytettävyyteen ja tämä on ollut suunnannäyttäjä Devise, Investigate ja Improve menetelmää kehitettäessä. DII- menetelmä kokonaisuudessaan on suunnittelua, tutkimusta ja parantamista. Sillä on pääajatuksena parantaa käytettävyyttä mutta myös opettaa käyttäjäänsä tarkastelemaan omaa työtään kriittisesti jo kehitysvaiheessa, ja tätä kautta panostamaan käytettävyyteen ja oppimisen myötä jopa välttämään virheitä. Menetelmän malli perustuu aktiiviseen käyttöön ja kommunikointiin tiimin jäsenten kesken ja systemaattiseen paneutumiseen kuitenkin väljien rajojen kanssa, kuten ketterään testaukseen kuuluu.

Myöhemmässä vaiheessa on mahdollista, että menetelmän käyttöönottoa tai sen käytöstä saatavista tuloksista tehtäisiin tutkimusta esimerkiksi opinnäytetyön muodossa.

Avainsanat (asiasanat)
Käytettävyys, käytettäyytestaus, ketterät menetelmät

Muut tiedot

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

DESCRIPTION

Author(s)	Type of publication	Date
PUIKKONEN, Heli	Bachelor´s Thesis	21112011
	Pages 35	Language Finnish
	Confidential ( ) Until	Permission for web publication (X)

Title
USABILITY TESTING AS A PART OF AGILE SOFTWARE DEVELOPMENT

Degree Programme
Business Information Systems

Tutor(s)
Kiviaho, Niko

Assigned by
Polar Electro Oy

Abstract
This thesis, Usability testing as a part of agile software development was assigned by Polar Electro Oy. The main pbjective pf the thesis was to develop a procedure that is to be a part of everyday usability testing in a software development project in accordance with the company's needs. The purpose of the procedure is to ease the improving of usability, particularly before the final completed product is released.

The research was carried out as a survey to get the start-up point of the company. The respondents were from several teams and segments and they consisted of specialists. The most important issue that came out was that the company does not have any specified and consistent method for usability measuring in software development projects. When the theory part was created it came out that it is possibly to affect the usability a great deal with only resources. This has been a guide line when Devise, Investigate and Improve procedure was created. The DII- procedure helps the users to assess their own work and hopefully realize how to avoid the problems during the learning as early as in the design phase. The model of the procedure is based on active communication and used within the team members who need a systematic concentration on the task hand. Everything takes place within loose lines, as it should in agile testing. At a later stage it is possible to research how the assignment has succeeded, and also study what the results of research coul be when using the procedure.

Keywords
Usability, usability testing, agile method

Miscellaneous

SISÄLTÖ

1 JOHDANTO

Käytettävyys. Ohjalmistosuunnittelun ja –tuotannon näkökulmasta tämä varmasti on tärkeysjärjestyksessä melko korkealla projektin aikana, aivan kuten käyttäjällekin käyttökokemuksen aikana. On oletettavaa, ettei käyttäjä halua sen paremmin käyttää kodinkonetta joka on käytettävyydeltään haasteellinen kuin web-sivustoa, -palvelua tai –sovellustakaan. Steve Krug (2006, 11) kirjassaan ”Don’t make me think!” kirjan nimen mukaisesti tuo oivaltavasti esille, että olisi tärkeää, ettei käyttäjää tarvitsisi pakottaa ajattelemaan. Kuinka ideaalia tämä olisikaan, mutta kuinka vähän se oikeassa elämässä toteutuu. Jokainen meistä on varmsti törmännyt ongelmiin erilaisia internetpalveluita ja –sovelluksia käyttäessään. Miksi tällaisia ongelmia sitten esiintyy, vaikka työ tehtäisiin huolellisesti? Ensimmäinen asia, joka tulee mieleen ihan jo maalaisjärjelläkin ajateltuna on se, että kun on kyse ihmisten tekemästä ja luomasta asiasta, on inhimillisen erehdyksen ja virheen mahdollisuus aina olemassa. Toinen tärkeä huomio on se, että suunnittelussa on voinut tapahtua virhe, mutta myös testaus on voinut olla puutteellista. Tästä pääsemmekin kätevästi aiheeseen: Käytettävyystestaus osana ketterää ohjelmistokehitystä.

Omalta osaltani olen enemmän kuin tyytyväinen opinnäytetyöni aiheeseen ja siihen, että saan varmasti tulevaisuutta varten osaamista ja näkökulmia prosessin aikana. Yhteistyö Polar Electro Oy:n kanssa on hieno mahdollisuus kehittää ohjelmistotuotannon mutta myös käytettävyyden saralla osaamistani, kuuluvathan ne tälläkin hetkellä jokapäiväiseen työskentelyyni käyttöliittymäsuunnittelijan työssäni. Näin ollen se yhteys ei ole motivaattoreista pienin. Tarkoituksenani on tuottaa menetelmä, joka tukee päivittäistä käytettävyystestausta osana joka päiväistä työskentelyä ohjelmistotuotannon parissa juuri Polar Electro Oy:n ohjelmistotuotantoprojekteissa. Uskon, että opinnäytetyöni yhdessä työharjoitteluni kanssa rautaisten ammattilaisten ohjaamina auttavat minuakin pääsemään tavoitteeseeni; oleman yksi heistä jonakin päivänä.

2 TUTKIMUSASETELMA

2.1 Tutkimuksen taustaa, tavoitteet ja rajaukset

Tavoitteena on kehittää menetelmä, jolla on mahdollista hyödyntää jokapäiväistä ketterää testausta järkevänä osana iteratiivista ohjelmistokehitystä ja jonka avulla voitaisiin maksimoida virheiden löytäminen jo ennen, kuin ne menevät implementointiasteelle mutta toisaalta myös implementoinnin jälkeen ennen kuin tuote tuodaan loppukäyttäjätestauksen ulottuville. Jakob Nielsenin tunnetuksi tuomat heuristiikat käytettävyydestä (http://www.useit.com/papers/heuristic/heuristic_list.html 29.8.2011) tulevat olemaan opinnäytetyössä osana menetelmän kehittämistä ja kyselytutkimuksen analysointia, mutta asioita pyritään tarkastelemaan myös muiden asiantuntijoiden näkemyksiä hyväksikäyttäen, sekä pyrkimällä luomaan uusia innovaatioita. Oletuksena kuitenkin on tuoda näkökulmia ja mahdollisuuksia päivittäiseen käytettävyystestaukseen unohtamatta kyselyä ja tulosten analyysia siitä, mikä yrityksen käytettävyystestauksen lähtötilanne on ja kuinka tiimin jäsenet kokevat päivittäisen ketterän testauksen osana omaa panostaan projektin eri kehitysvaiheissa. Tarkoituksena on tarkastella samalla myös miksi jokapäiväistä ketterää käytettävyystestausta saatetaan vierastaa ja katsoa testauksen hyötyjä mutta myös mahdollisia haittoja projektin kehityksen näkökulmasta siten, että kyseessä on kuitenkin asiakkaalle tuotettava palvelu, jonka käyttäminen täytyy kokemuksena olla mahdollisimman positiivinen. (Krug 2006, 11)

Ketterälle käytettävyystestaukselle tulisi tulisi asettaa tavoitteet ja rajat, joiden kautta pyritään selvittämään, mitä tuloksilla halutaan hakea ja millaista hyötyä niistä pyritään saamaan. On kuitenkin järkevää olla realistinen saavutettujen tavoitteiden suhteen, mutta samalla huomioida millainen on saavutettu hyöty taas suhteessa kulutettuun aikaan ja muihin resursseihin. Totta on, että kun testauksella halutaan saavuttaa maksimaalinen hyöty, ei tuloksissa mahdollisesti ilmenneitä ongelmia voida vain sivuuttaa ja jättää huomiotta. Käytettävyystestauksen ottamisena osaksi jokapäiväistä

ketterää ohjelmistotuotantoprojektia suurin hyöty lienee juuri siinä, että reagointi virheisiin on mahdollista ajoissa. (Koskinen 2005, 189) Karkeana vetona voidaan todeta, että vesiputousmallilla ja ketterällä mallilla eräs suuri ero kehitysprojektissa löytyy testauksesta ja mahdollisuuksista reagoida ajoissa ilmenneisiin ongelmiin. Ketterän mallin projekteissa on helpompaa tehdä muutoksia ja reagoida riskeihin, ongelmiin ja kehityskohteisiin ripeämmin, korjaamaan ne lähes kivuttomasti ja jatkamaan tämän jälkeen projektia suunnitellusti iteraation loppuun asti. Vesiputousmallilla takaisinpalaaminen on aina vaikeampaa, kun kuitenkin projektia pyritään toteuttamaan kokonaisuutena askel askeleelta valmiiksi. Näin ollen kesken tekemisen ei ole yhtä työlästä nousta kaatumisen jälkeen ylos, kun toteuttetaan pala kerraallaan. (http://www.okol.org/verkkokurssit/datanomi/tietojarjestelmien_kaytto_ja_kehittaminen/johdatus_tietojarjestelmiin/kehittamistyon_vaiheet_ja_elikaarimallit/kehittamistyon_vaiheet_ja_elinkaarimallit_asia.htm 4.10.2011)

2.2 Tutkimus- ja kehittämismenetelmät

Opinnäytetyön toteutusmenetelmä on ensisijaisesti kvalitatiivinen, eli laadullinen menetelmäsuuntaus jonka päämääränä on ymmärtää tutkittavaa ilmiötä mahdollisimman hyvin. Tutkimusmenetelmänä käytetään kyselytutkimusta.

Laadullisen tutkimuksen erityispiirteisiin kuuluu, että tutkimushenkilöt, eli tässä tilanteessa Polar Electro Oy:n työntekijöistä koostuvan, valikoidun ja tarkoituksenmukaisen asiantuntijajoukon henkilöt vastaavat vapaamuotoisesti kyselyyn (liite 1.) ja saavat haluamansa mukaan kertoa kokemuksistaan ja mielipiteistään sekä käytettävyydestä ja testauksesta tällä hetkellä; toisin sanoen ihminen on tässä tapauksessa tiedonkeruun väline. (Hirsjärvi, Remes & Sajavaara 2006, 154) Kvalitatiivista tutkimusta kutsutaan myös ymmärtäväksi tutkimukseksi juuri sen ihmisystävällisyyden vuoksi. Kyseinen menetelmäsuuntaus on valittu siksi, että voidaan saada myös kartoitettua ymmärrystä käytettävyydestä ja käytettävyystestauksesta ja tätä kautta

päästä perille siitä, mikä on tilanne tällä hetkellä ja kuinka saadaan kehitettyä mahdollisimman järkevä malli arkipäivän testaamiseen lyhyidenkin iteraatiojaksojen aikana.

Kyselytutkimuksella ja sen analysoimisella pyritään tuomaan mukaan nykyisten ohjelmistokehitystiimin näkökulmat ja asenteet projektien puitteissa ja tämän kautta kehitysmahdollisuudet tulevaisuuden projekteja varten. Tilanteen analysoimiseen ja menetelmän kehittämiseen käytetään myös heuristisen läpikäynnin analysointimallin piirteitä. Tästä lisää luvussa 3.2.

Tutkimuksen validiteettia on pidetty silmällä juuri asiantuntijaryhmää valittaessa sekä kysymysten asettelussa.

2.3 Tutkimuskysymykset

Tutkimuskysymyksiksi nousivat suunnittelu- ja rajausprosessin aikana seuraavat:

1. Mitä on ketteriä menetelmiä käytettäessä käytettävyystestaus ?

Käytettävyystestaus ketteriä menetelmiä käytettäessä voidaan suunnata esimerkiksi ohjelmiston tutkimiseen tai käyttämiseen ja sen ymmärtämiseen ja tulkitsemiseen. Ohjelmistokonseptin ymmärtäminen on tärkeää testauksen onnistumisen kannalta. Testitapauksia ei yleensä suunnitella etukäteen eikä näin ollen ole etukäteen päätetty, mitä testataan. Tuotteen käyttäytymisen tutkiminen on tärkeää ja toiminnallisuuksien toimimiseen on hyvä kiinnittää huomiota. Ketterälle käytettävyystestaukselle ei useinkaan anneta muuta suuntaa kuin kehys, jonka sisällä suorittaa testaus. Arvioijan havaintoja kuitenkin olisi hyvä seurata häiritsemättä itse testausta. Testaustapojen muuttaminen on myös suotavaa aika ajoin tai jopa aina uusien havaintojen jälkeen, jotta saataisiin paljastettua enemmän virheitä. Kun testatessa tuotetaan usein uusia testitapauksia, tulisi ne ottaa myös mukaan päivitettäviin testisuunnitelmiin ja jakaa

tieto kaikille osapuolille. .
(http://mattivuori.net/julkaisuluettelo/liitteet/kettera_testaus.pdf 9.11.2011)

Käyttötapaus toimii usein suoraan lähtökohtana. Myös ennakkokäsitykset ovat jokseenkin tärkeitä; tärkeysjärjestys testattaessa toiminallisuuksia on helpompi muodostaa. Täytyy kuitenkin muistaa, että usein uudetkin innovaatiot ovat keksitty jo aiemmin ja tästä syystä olemassa olevaa materiaalia olisi järkevä hyödyntää tutkimalla esimerkiksi kilpailijan tuotetteita. Tärkeimmät ketterän testauksen piirteet ovat, että testauksen tuottama uusi tieto aiheuttaisi reagointia ja muutosta parempaan, mutta myös ymmärrys siitä, ettei kaikki mene aina ennakkosuunnitelmien mukaan. Laatua rakennetaan pala palalta ketterässä ohjelmistokehityksessä ja teknisen laadun tulisi olla jatkuvasti kunnossa, testauslähtöisyys on tähän yksi keino ja näin myös riskitason tulisi pysyä alhaisena. (http://mattivuori.net/julkaisuluettelo/liitteet/kettera_testaus.pdf 9.11.2011)

2. Kuinka projekteissa toimitaan tällä hetkellä?

Tällä hetkellä projektien testaus on laskettu projektin laajuuteen, eli scopeen. Testausosaston vastuulla on toiminnallisten testien tekeminen suunnitellusti ja aikataulussa. Yleensä testaaminen tapahtuu aina, kun jokin osio valmistuu. Löytyneet virheet kirjataan ylös ja korjataan sen mukaan, mikä on tarve ja ajallinen tilanne. Myös loppukäyttäjätestausta on liitetty projektiin ja tapahtuu pääasiallisesti projektin loppupäässä, jolloin voidaan parhaiten kartoittaa testausta mahdollisesti eniten tarvitseva osio tai testata selkeästi koko tuotetta. Loppukäyttäjätestauksen tulokset analysoidaan ja onkin pilotoitu, että tulosten jälkeen varataan aikaa tarvittaville korjauksille tuotteeseen.

3. Kuinka saadaan yhdistettyä ketterien menetelmien käytettävyystestaus osaksi yrityksen toimintamallia ja tehostettua nykyistä testaamista?

Tärkeimpänä keinona jo ketterää testausmallia osittain käyttävien projektien testauksen tehostamiseen ja entistä ketterämmäksi siirtymiseen on varmasti se, että testaukselle

jätetään sitkeästi vain enemmän tilaa, kun scopea suunnitellaan ja luodaan. Kyse ei ole siitä, että sille tarvitsisi suuria määriä varata aikaa, vaan lähinnä juuri useisiin eri vaiheisiin projektin kulkua. Pienelläkin ajalla usein käytettynä voidaan varmasti tehostaa käytettävyyttä ja laatua. Koska ketterän testauksen piirteitä projekteissa löytyy jo nyt, ei muutos varmastikaan olisi suuri tai kuormittava ajatellen ihmisiä, jotka projektissa ovat. Muutokset kuitenkin näiden asioiden kannalta ovat toivottavasti aina tervetulleita vaikka vaikuttavatkin tiukkoihin aikatauluihin todennäköisesti kiristävästi. Tarkoituksena olisikin vakiinuttaa menetelmän kanssa projektin sisältöön suurempi määrä testaamista ja panostusta käytettävyyteen myös selkeästi koko tiimin osalta.

3 KÄYTETTÄVYYS JA HEURISTINEN LÄPIKÄYNTI

3.1 Mitä on käytettävyys?

Käytettävyys on terminä hieman vaikeaselkoinen ja hankalasti avattava. Sille kuitenkin on olemassa oma ISO13407-määritelmänsä, jonka mukaan käytettävyys koostuu kolmesta osatekijästä; tarkkuus, tehokkuus ja tyytyväisyys jotka toimivat mittareina siitä, ovatko määritellyt tavoitteet saavutettu käyttäjien toimesta. (http://ww.cs.tut-fi/~sevi/luennot/seviluennot4.pdf 9.11.2011) Sen voisi sanoa olevan myös tuotteen soveltuvuutta tietylle käyttäjälle tiettyyn tehtävään. Nämä yksinkertaisuudessaan kertovat jo enemmän kuin itse termi, mutta Nielsenin laajalti kuvaava, selkeä ja kattava määritelmä käytettävyydesta kertonee vielä hieman enemmän. Se pitää sisällään viisi erilaista arttribuuttia, eli laatukomponenttia. Näitä ovat opittavuus, tehokkuus, muistettavuus, virheettömyys ja tyytyväisyys. Näiden avulla pyritään takaamaan, että tuotteen käyttäminen olisi selkeää ja tehokasta ja sen käyttäminen olisi helppo opetella. Myös virhetoimintojen minimaalinen määrä on käyttäjälle tärkeää, ja itse käyttäjäkokemus tulisi tehdä miellyttäväksi käyttäjälle. (1993, 26-35)

Steve Krug lähtee selkeästi siis samoilta linjoilta käytettävyyteen kirjassaan kirjassaan Don’t make me think! (2006, 11), kuin Nielsen aiemmin; käyttäjälle tehtäisiin asiat mahdollisimman helpoiksi. Nielsenin mukaan sivuston rakenteella on suurempi merkitys käytettävyyden kannalta kuin yksittäisen sivun designilla, mutta suunnittelijoiden tulisi kuitenkin muistaa, että kaikesta huolimatta käyttäjä kontrolloi liikkumastaan palvelussa itse ja tekee myös lopulta itse ratkaisunsa. (2000, 11-22) Virheitä palvelussa kuitenkaan eivät tee käyttäjät vaan suunnittelijat.

Eräs toinen tunnettu käytettävyyden määrittelijä on Ben Shneidermann, jonka kehittämät kahdeksan kultaista sääntöä auttavat myös ymmärtämään käytettävyyttä, mutta myös käyttöliittymäsuunnittelua (liite 2.). Hänenkin periaatteensa käytettävyydesta ovat selkeys ja helppokäyttöisyys, kuten Nielsenillä ja Krugillakin yllä. Niin Shneidermannin kuin Nielseninkin listauksissa on kuitenkin yhteneväisinä

attribuutteina useita asioita, näistä muun muassa virheistä toipumisen molemmat luokittelevat sellaiseksi, joka täytyy tehdä käyttäjälle mahdollisimma kivuttomaksi ja helpoksi. Samoin on käyttäjän muistin kuormittamisen laita, jota molemmat suosittelevat varomaan ja kontrolloimisen tunne, joita molemmat pitävät tärkeänä asiana käyttäjän itsenäisen toimimisen tukemiseksi.
(http://www.cs.tut.fi/~grako/2008/luennot/grako_kayt_luento.pdf 4.11.2011,
http://www.useit.com/papers/heuristic/heuristic_list.html
29.8.2011)

3.2 Käytettävyysheuristiikat ja -testaus

Jakob Nielsen on kehittänyt myös kokoelman sääntöjä ja ohjeita käytettävyydestä, näitä kutsutaan käytettävyysheuristiikoiksi (liite 3.). Nämä mahdollistavat tuotteen arvioinnin jo varhaisessakin kehitysvaiheessa. Nielsenin heuristiikat pitävät sisällään hieman samoja asioita, kuin hänen jo edellä mainittu käytettävyyden määritelmänsä, mutta sisältää myös lisäyksiä. Heuristiikkoja mukaileva kehitysmalli, heuristinen läpikäynti on Nielsenin mukaan nopea ja edullinen toteuttaa ja hänen mukaansa yhden arvioijan löydos voi olla jopa 35% ongelmista, joita tutkittavassa kohteessa esiintyy. Näin ollen jo kolme arvioijaa olisi riittävä, mutta viisi olisi hänen mukaansa sopivin ja tehokkain arvioijamäärä parhaan mahdollisen lopputuloksen saavuttamiseksi. Jokaisen arvioijan löytämät ongelmat voivat olla keskenään erilaisia ja viiden arvioijan tuotos on jo mittava ongelmien löytymisen ja eriäväisyyksien näkökulmasta. (1993, 156)

Käytettävyyden arviointi- ja analysointimenetelmänä heuristinen läpikäynti on käytännössä asiantuntija-arvio käytettävyydestä. Samalla sen vahvuus voi osoittautua myös heikkoudeksi, sillä kun mukana on vain ammattilaisia ja asiantuntijoita, saattaa näkökulma olla juuri ammattimainen ja monisäikeisempi, kun tietoa ja ymmärrystä voi asiantuntijoilla olla enemmän kuin loppukäyttäjällä. Tämä on kuitenkin käytetty ja joustava menetelmä, ei varmastikaan vähiten tehokkuutensa vuoksi. (s. 156)

Käytettävyystestauksella on selkeästi kasvavassa määrin painoarvoa palveluiden suunnittelussa, ohjelmistotuotannossa ja –kehityksessä. Aikaisessa vaiheessa tehdyn testaamisen hyötyjä painottaa muun muassa Steve Krug. (2006, 134) Jos testataan yhdelläkin arvioijalla aikaisessa vaiheessa, ovat hyödyt hänen mukaansa suuremmat kuin että testattaisiin 50:llä arvioijalla loppuvaiheessa. Samalla täytyy muistaa, että testauksen tavoitteena ei ole todistaa jotakin ominaisuutta tai toiminnallisuutta hyväksi tai huonoksi, vaan pääasia on löytää ja paikallistaa ongelmat. Tekijän itsensä suorittama testaus on harvoin riittävän objektiivista, sillä hän tietää tuotoksestaan liikaa, ei pysty ajattelemaan täysin ulkopuolisen silmin. (2006,133-135)

On tärkeää, että ainakin kilpailijan tuotteet tai palvelut ovat selvillä. On turha olla hyödyntämättä jo käytössä olevaa materiaalia ja suuntaa siitä, miten käyttäjät osaavat toimia. Suuria määriä erilaisia resursseja on kuitenkin jo käytetty erilaisiin tuotteisiin ja ”ilmaista materiaalia” jonkun toisen tutkimana ja selvittämänä löytyy pelkästään internetiä selaamalla käsittämättömiä määriä. Testauksessa testattavan systeemin ei tarvitse olla edes kokonaan valmis vaan riittää, että edes prototyyppi tai toiminnallisuudet ovat paikoillaan. Jopa tässä vaiheessa joidenkin käytettävyysongelmien löytyminen on mahdollista. Ensin täytyy kuitenkin päättää mitä arvioijille tarjotaan, ja kuinka usein. Myös on hyvä päättää millaisia toimintoja tai osa-alueita testataan, vai testataanko koko palvelu. Jotakin suuntaa kannattaa asettaa myös sille, millaisia ongelmia korjataan ja millaisten annetaan olla mahdollisesti seuraavaan projektiin asti, jonka aikatauluun ne saattavat mahtua paremmin. Jokaisella yrityksellä on omat resurssinsa testata niin ajallisesti kuin rahallisestikin ja myös korjata tahtoessaan palvelunsa samoin kuin jokaisella ohjelmistolla ja palvelulla on omat käytettävyysongelmansa, mutta luonnollisesti vakavimpien ongelmien selvittämiseen tulisi paneutua ensin. Krug puhuu ”Rocket surgery made easy”- kirjassaan (2010, 43) samaa kieltä Jakob Nielsenin kanssa siitä, että jo kolmella arvioijalla saavutetaan hyviä tuloksia edullisesti, kun hyödynnetään testausta alusta lähtien ja panostetaan virheiden korjaamiseen.

4 KETTERÄT ELI AGILE-MENETELMÄT

Ketterät eli agile- menetelmät ovat ohjelmistotuotannon menetelmiä, jotka helpottavat jakamaan yrityksen rajallisia resursseja oikein ja joilla saadaan tehokkuutta työskentelyyn sekä vältettyä turhaa työtä. Yleistyksenä agile-menetelmät kuvaavat joukkoa menetelmiä, joissa palautteen saaminen nopeasti tehdystä työstä on projektin yksi avaintekijä ja auttaa täten projektia pysymään oikeilla urilla. Monet ketterien menetelmien tiimeistä ovat itseorganisoituvia, mutta eivät kaikki. On myös mahdollista, että tiimillä on vain nimellinen projektipäällikkö huolehtimassa projektin kulusta. Ei ole olemassa yhtä oikeaa ja toimivaa menetelmää ketterien joukosta, joka sopisi jokaiselle yritykselle. Menetelmät vaativat aina sovittamista yrityksen prosessimalliin, ja mahdollisesti eri variaatioiden yhdistelmästä saadaan rakennettua kullekin omansa ja parhaiten soveltuva vaihtoehto. Riskien hallintaa pystytään kartoittamaan kattavasti, sillä uusien toiminnallisuuksien esittelysykli on nopea ja tieto jaetaan mahdollisimman nopeasti osion valmistumisen jälkeen muulle prokjektijäsenistölle. Näin ollen koko tuotteen ennustettavuus ja läpinäkyvyys paranevat huomattavasti ja reagointi riskeihin ja niiden mahdollisiin vaikutuksiin helpottuu. Myös liiketoiminnan edustajan jatkuva läsnäolo projektin kulussa on tärkeä osa tarpeiden huomioimisessa mutta myös näkemysten ja palautteen saamisessa. Kommunikaation merkitys ketterissä menetelmissä on todella suuressa osassa koko projektia ja saattaa vaatia töitä sekä asennemuutoksia organisaation sisällä toteutuakseen. Ketterät menetelmät myös mahdollistavat alati muuttuvassa ohjelmistokehitysympäristössä toimimisen vaivattomasti sekä suuntaamaan resurssit tarpeen tullen juuri sinne, missä niiden hyöty voidaan maksimoida. Tällaisia menetelmiä käyttävät myös muun muassa Google ja Facebook, sekä Microsoft. (http://reaktor.fi/osaaminen/agile/ 4.11.2011)

5 TUTKIMUSTULOSTEN ANALYSOINTI JA LÄHTÖTILA

Tutkimusta ja lähtötilakartoitusta varten lähetettiin kysely 14:lle asiantuntijalle. Kartoitus on tehty kaikkien 9:n palauttaneen vastauksia hyödyntäen. Asiantuntijajoukkoa edustavat kyselyyn vastanneissa muun muassa team leader, UI-designer, systems designer, usability designer, testing designer sekä user experience validation manager.

Yrityksen käytettävyyden ja käytettävyystestauksen tilan kartoittamisen yhteydessä kävi kyselystä selväksi, ettei joka päiväisen käytettävyystestauksen mallia ohjelmistotuotannon puolella ole valmiina. Tutkimuksesta ilmeni myös, että osa vastanneista oli hyvinkin selvillä millaisia käytänteitä yrityksellä on käytössä, suurin osa rannelaiteprojektien puolella. Kaikilla ei kuitenkaan ollut selkeää käsitystä siitä, miten käytettävyyden analysointi toimii. Vastausten perusteella koetaan, että selkeästi enemmän kuitenkin panostetaan rannelaitteisiin keskittyneellä puolella kuin ohjelmistokehityspuolella. Kävi ilmi myös, että käytettävyyden analysoinnin koetaan olevan kokonaisuudessaan melko pienten resurssien varassa, mikäli sitä ollenkaan edes kartoitetaan. Sen tarpeellisuus on selkeästikin ilmassa ja se tiedostetaan, mutta sen toteuttaminen syystä tai toisesta useimmiten jää. Vastauksista selvisi myös, että käytettävyystestausta hoidetaan enemmän juuri rannelaitepuolella, kun taas ohjelmistotuotannossa sen hyödyntäminen on harvinaisempaa, mutta onneksi kasvamassa. Aineiston mukaan alkuvaiheessa on mahdollista kuitenkin testata käytettävyyttä esimerkiksi käymällä edes paperilla protoversioita lävitse tai mallinnusohjelmalla tehtyjä mockuppeja, eli mallinnuksia. Tätä kautta saadaan jonkinlainen kartoitus käytettävyydestä ja sen tarpeesta, sen jälkeen kun kohderyhmä ja sen asettamat tarpeet on kartoitettu. Loppuvaiheen käytettävyystestaus on niinikään yleisempää kuin selkeästi koko projektin elinkaaren aikana tapahtuva testaus.

Myös tiimien kesken on vastauksissa huomattavissa eroja, eniten juuri rannelaite- ja ohjelmistotuotantotiimien välillä. Ohjelmistopuolella ollaan kuitenkin kallistumassa koko

ajan kohti ketterämpiä malleja kun taas tuntuu, että rannelaitepuolella suuntaus ei ole niinkään voimakas. Näiden kahden projektin yhteensovittamisen yhdeksi ongelmista koetaan se, että systeemituotteeksi kutsuttu kokonaisuus koostuu kahdesta erillään toimivasta, itsenäisestä projektista, jotka usein valmistuvat eri aikaan. Tällaisen systeemituotteen, rannelaitteen ja ohjelmistopalvelun käytettävyydestä saadaan kattava kuva kuitenkin vasta kun molemmat ovat valmiita tai lähes valmiita ja tässä vaiheessa käytettävyystestauksen aloittaminen on auttamatta myöhässä. Moni toivoikin, että kuilu ei ainakaan kasvaisi enempää eri tiimien välillä, jotta saataisiin minimoitua erillään tekemista.

Projektien usein tiukaksi asetetut aikatauluat tavoitteineen tuntuvat olevan joidenkin vastanneiden mielestä suuri kompastuskivi, jonka vaikutukset kohdistuvat vahvimmin valitettavasti juuri käytettävyyspuoleen ja sen testaamiseen. Ilmoille heitettiin jopa, etteivät projektipäälliköt voi tai edes halua antaa tiiviistä aikatauluistaan riittävästi tilaa käytettävyyden kartoittamiseen, parantamiseen ja testaamiseen tai, että esiin tulleet bugit ja enhancementit, kehitysideat sivuutetaan ja jätetään pahimmassa tapauksessa roikkumaan ja odottamaan seuraavaa projektia, elleivät näiden prioriteetit ole riittävän korkeat välittömään korjaukseen. Vastauksista oli havaittavissa myös, että joskus projektipäälliköiden asennetta parannusehdotuksiin pidettiin turhan jarruttavana ja henkilökohtaista motivaatiotaan alentavina. Myöskään vastuuhenkilöt eivät olleet selvät juuri käytettävyyden osalta, tai koettin, että sellaista ei ehkä ole nimetty tehtävään erikseen lainkaan. Tuli myös ilmi ehdotuksena, että UI-designerin vastuulle asetettuna tämä saatettaisiin saada hoidettua kätevimmin, sillä hän kuitenkin on suunnitellut käytettävyyden ja hän on henkilö, joka osaa perustella, varsinkin testitulosten kera ratkaisunsa myös mahdollisesti vastassa oleville skeptikoille. Toiveena kaiken kaikkiaan tuntui olevan, että tämä asia selkiytyisi, jotta oman roolin ja toimenkuvan kokonaisuus olisi itselle selkeämpi ja epätietoisuudesta päästäisiin.

Useiden vastausten joukosta tuli kuitenkin esille se, että käytettävyyden mittaamista ja käytettävyystestausta tulisi parantaa ja tuoda enemmän osaksi projektia. Asiakastyytyväisyyden nostaminen on varmasti yksi tärkeä asia kaikissa muissakin, kuin

ohjelmistotuotannon projekteissa. Kaikesta huolimatta yleinen asenne ja vastaanotto ketterään ohjelmistokehitykseen siirtymisestä tuntuu olevan positiivinen ja jokainen vastaaja korosti käytettävyystestauksen merkitystä kehitysprosessin aikana, olipa kyse sitten minkä tason asiantuntijasta tahansa.

6 DEVISE, INVESTIGATE AND IMPROVE- MENETELMÄ

Kuten edellä on jo mainittu, on menetelmä tarkoitus osoittaa juuri ohjelmistokehityksen puolelle ottamatta sen suuremmin kantaa rannelaiteprosessien toimintamalleihin. Agile-menetelmistä on nähtävissä lisääntynyt kommunikaatio projektiryhmän jäsenten välillä sekä lyhentyneet uusien toiminnallisuuksien ja osioiden esittelyt sekä se, että myös tuoteomistajat ovat pääasiallisesti aina läsnä suunnittelussa, katselmoinneissa ja statuspalavereissa ja ottavat kantaa tarpeen mukaan ja tuovat kehitysehdotuksia tarpeisiinsa. Iteraatiojako projektin aikataulussa tukee agile- menetelmien mallia.

Projektit ovat siis jaettu iteraatioihin, joiden keskimääräinen kesto on noin kaksi viikkoa kerrallaan. Testausosasto on mukana myös jatkuvasti, ja pyrkii testaamaan mahdollisten bugien varalta jo tuotettua ohjelmistoa. Bugit raportoidaan projektiryhmälle, erityisesti kehittäjille mahdollisten korjausten aikaansaamiseksi mahdollisimman nopeasti testauksen päätyttyä kyseisen osion kohdalta. Kuitenkaan projektin scopeen ei ole laskettu muulle, kuin testiosaston testaukselle sekä loppukäyttäjätestauselle aikaa. Työkalulla pyritään myös saamaan varattua paremmin aikaa jokaiseen iteraatioon ja siinä tapahtuvaan testaamiseen vielä aktiivisemmin.

Pohjana menetelmälle on ollut perinteinen agile- menetelmien kaava, jonka neljä arvoa prosessissa ovat suunnittelu, toteutus, testaus ja hyväksyntä, ja jotka kulkevat läpi koko prosessin, jokaisen iteraation ja jokaisen osion toteutuksen vuorollaan. (Liite 4.) Myös Willim Demingin PDCA- menetelmä on tukenut jokapäiväisen käytettävyystestausmenetelmän kehittämistyötä ja antanut suuntaa, kuinka prosessin sisällä ketterästä testauksesta saataisiin kaikkein eniten irti, jokaisessa projektissa sovelletusti. (http://laatu.tkk.fi/fi/toimintakasikirja/pdca-plan-do-check-act.pdf) Plan, Do, Check ja Act on tapa, jolla laadunvarmistus voidaan hoitaa systemaattisesti ja joka perustuu jatkuvaan parantamiseen. Kuten agile-menetelmien kaava, niin PDCA:kin perustuu siihen, että tapa toimia on jatkumo eikä kertaluontoinen malli.

Kehitetty menetelmä on malli, joka ei astu ketterän mallin varpaille tai kilpaile sen

kanssa, vaan on omiaan tukemaan kehitystä ja laadunvarmistusta. Devise, Investigate, Improve, jatkossa DII (Liite 5.), on malli jonka avulla pyritään tekemään yrityksen ohjelmistokehitysprojektien arvoketjuissa laadukkaampia tuotteita vaikuttamatta itse projektin aikatauluun venyttävästi. Käytettävyyden parantaminen on ensisijainen näkökulma josta lähdetään liikkeelle, tällöin tavoitteiden täytyy olla projektin sisällä selvät ja sovitut, ja on toivottavaa, että prosessissa mukana oppii niin yksittäinen työntekijä, mutta myöhemmässä vaiheessa myös organisaatio, olettaen, että menetelmän avulla on saatu positiivisia tuloksia aikaan. DII:n käyttämisessä on tärkeä muistaa, että jokainen osio menetelmästä tukee toista. Toisin sanoen mikään osista ei yksinään riitä parantamaan käytettävyyttä, vaan kaikki kolme on käytävä läpi, jotta päästäiaiin lopputulokseen, joka on tavoiteltu.

Menetelmän tarkoitus on helpottaa projektiryhmää, erityisesti suunnittelijoita ja kehittäjiä hyödyntämään testausta osana jokapäiväistä työskentelyään. Tahtoessaan vaikka joka päivä sovelletusti. Menetelmä ei kuitenkaan sulje missään vaiheessa pois testausosaston suorittamaan testausta tai loppukäyttäjätestausta, tai muita mahdollisia tarvittavia menetelmiä, joiden avulla voidaan varmistaa käytettävyydelle mahdollisimman korkea taso. Olisi toivottavaa, että mahdollinen muutosvastarinta olisi helposti murrettavissa ja käytettävyyden parantamiseksi oltaisiin valmiita tekemään muutoksia vanhoihin, tuttuihin toimintamalleihin.

6.1 Menetelmän periaatteet

Ensimmäiseksi asiaksi kolmivaiheiselle DII- menetelmälle ketterän ja ketterähkön ohjelmistokehitystiimin projektissa luokitellaan jo sovittuun aikatauluun riittävä määrä ajallisia resursseja. Mikäli projektin aikatauluun ei sovitusti saa järjestettyä käytettävyystestaukselle sen vaatimaa ja asettamaa aikaa ja resurssia, on testaus organisoitava jotenkin muuten tai jatkettava toiminnallisella ja loppukäyttäjätestauksella. Ilman ketterän testauksen asettamista tärkeysjärjestyksessä tarpeeksi korkealle on sen suorittaminen tarvittavan tehokkaasti hankalaa. Ajatuksena olisikin, että työajanseurantajärjestelmiin laitettaisiin myös jatkossa omat taskinsa, eli työtehtävänsä henkilöresursseineen ylös jo varhaisessa vaiheessa ja suunnittelussa otetaan huomioon testauksen asettamat aikaresurssit ja nämä sisällytetään scopeen. Näin ollen sekä tiimi, että testauksesta huolehtiva ihminen ovat tietoisia vastuualueistaan ja työtehtävän vaiheesta ja niiden edistymistä on madollista tarkkailla aktiivisesti. Luonnollisesti tämä vaatii oman panoksesa jokaiselta tiimnin jäseneltä ja vaatii sitoutumista yhteisten pelisääntöjen toteutumiseksi. Jokaisen iteraation päätteeksi olisi suotavaa varata aika tutkimiselle ja sen tulosten analysoinnille. Tähän ei ole tarkoitusta uhrata aikaa loputtomiin, vaan tehdä karkea veto ja silmäys käytettävyysratkaisuihin sekä kartoittaa miten eteneminen on järkevintä.

Toinen tärkeä asia on vastuuhenkilö. On järkevää, että alusta asti joku on nimetty tehtävään. Tätä kyselyyn vastanneistakin jotkut toivoivat epäselvyyksien välttämiseksi. Projektipäällikkö yhdessä esimerkiksi käyttöliittymäsuunnittelijan kanssa olisi varmasti paras vaihtoehto vastaamaan tehtävästä, tokikin siten, että projektipäällikön vastuu olisi ajallisesta järjestämisestä enemmän kuin itse testauksen vastuusta. Käyttöliittymäsuunnittelija tietää ratkaisunsa, osaa toivottavasti myös perustella ne ja näin ollen hyötyy varmasti eniten testien tuloksista jo mallinnus- ja suunnitteluvaiheessa, ennen kuin suunnitelmat menevät toteutukseen ja myöhemmin tuotantoon. Ajatuksena kuitenkin on se, että käyttöliittymäsuunnittelija olisi jatkuvasti askeleen tuotantoa edellä, aivan kuten hänen täytyy olla selkeästi läsnä niin projektin alussa, keskivaiheilla kuin lopussakin. Tästä johtuen testattavana ei aluksi olisi puhtaalta

pohjalta lähdettäessä muuta kuin käyttöliittymäsuunnittelijan mallinnukset esimerkiksi paperiprototyyppeinä. Iteraatioiden sisällön muuttuessa näiden paperiprotojen ohessa voitaisiin alkaa vähitellen testaamaan HTML-pohjaa ja myöhemmin kuvioon astuisivat jo implementoidut osat.

Kolmas huomionarvoinen seikka on ajankohta. Lähtökohta on se, että kun projekti toimii iteraatiomallilla, on jokaisen iteraation päätteeksi testaus sen aikana valmistuneiden osien osalta. Vähintäänkin yhtä monta kertaa tutkitaan, kuin on iteraatioita ja jokaiseen iteraatioon tulisi varata edes hieman aikaa siihen, että mahdolliset virheet, ainakin suurimmat ja kriittisimmät korjataan jo saman tien, eikä jätetä loppuun tai myöhemmäksi. Vaikka aikaa käytettäisiin tässä kohtaa virheiden korjaamiseen tai ongelmien ratkaisemiseen, on se aina jossain määrin pois projektin lopusta. On oletettavaa, että jos valmistaudutaan korjaamaan enemmän ja virheitä onkin vähemmän kuin odotetaan, eivät hommat projektin osalta lopu varmasti monessakaan tilanteessa kesken vaan vapaaksi jääneelle ajalle löytyy käyttöä projektin puitteissa. Ei ole huono asia, että virheiden vähyydestä johtuvaa yliaikaa käytettäisiin ylimääräisten osioiden tekemiseen vaikkeivat ne juuri tämän projektin suunnitelmiin kuuluisikaan, aina kaikki ylimääräinen on kuitenkin plussaa. Asiakastakaan tuskin haittaa, että käytettävyyden laatua pyritään parantamaan systemaattisesti koko projektin iteraatioiden ajan.

Neljäntenä asiana esiin nousevat testauksessa käytettävät keinot. Kuten edellä jo mainittiin, ensimmäisenä testauksen alle joutuvat paperiversiot mallinnuksista. Mallinnusohjelman sisälläkin saattaa olla mahdollista linkittää tiedostot toisiinsa, joten joissakin tilanteissa voidaan testata rautalankamallinnukset jo koneelta käsin. Tässäkin kohtaa ongelmien esiin nouseminen on varmasti täysin mahdollista, joten tällaista ei kannata väheksyä arvoltaan. Seuraavassa vaiheessa pitäisi olla mukana rautalankamallinnusten lisäksi myös HTML- versio ainakin päätoiminnallisuuksien kanssa valmiina testattavaksi. Tällaisesta saa kuitenkin jo enemmän selvää ja tuntua käytettävyydestä. Rautalankamallinnuksissa käytettävyys ei konkretisoidu samalla tavalla kuin jo HTML- versiossa, vaikka käytettävyyskonventioiden yhdenmukaisuudesta tässäkin voidaan huoli pitää. Lopulta testauksessa on mukana myös edelliseen

iteraatioon kuulunut implementoitu osio, jossa muun muassa linkitykset ja navigoimiset alkaisivat olla jo paikoillaan. Eli, testauskerralla x testataan rautalankamockuppeja, jotka tulevat seuraavan iteraation implementointiin ja juuri loppuneen iteraation aikana implementoituja oikeita osioita oikealla HTML- ulkoasulla. Näin varmistetaan se, että käyttöliittymäsuunnittelija tosiaan on askeleen edellä kehittäjiä, mutta myös kaikkien eri vaiheiden päätyminen testattavaksi sekä rautalankoina, HTML-versioina että implementoituna. Jokaista aluetta ja osaa testataan näin ollen vähintäänkin kolme kertaa ennen kuin tuote on edes kokonaan valmis. Kuitenkin ajankäytöllisesti mielipiteen tai muutaman kohdan läpikäyminen kerrallaan ei testauksen suorittajalta vie kuin muutaman minuutin, joka on hetkellisesti hyvin pieni uhraus projektin resursseissa. Tästä ei koidu kenellekään yhtä suurta haittaa kuin suhteessa projekti pitkässä juoksussa todennäköisesti ja toivottavasti hyötyy. Asiantuntijamielipidettä voidaan kysyä oman tiimin sisältä, mutta kannattaa pitää mielessä myös asiasta tietämättömät suorittajat, joiden mielipiteen tulisi painaa vaakakupissa melkoisesti kun suunnitellaan parempaa käytettävyyttä. Myös testauttaminen tuotteen omistajalla on suotavaa.

6.2 Devise

Devise- vaihe eli suunnittelu on lähtökohtaisesti tärkein vaihe. Se pitää sisällään kaksi eri suunnitteluvaihetta; projektin aikataulu- ja testaussuunnittelun DII:n puitteissa mutta myös itse käyttöliittymäsuunnittelun. Käytettävyyden arvionti lähtee DII:n osalta juuri tästä suunnitteluvaiheesta. Pääasiallisesti Devise kuitenkin tarkoittaa juuri käyttöliittymän suunnittelua, mutta sen voi laskea pitävän sisällään myös sen, että suunnitellaan sopivat paikat, henkilöt ja ajankohdat tutkimusvaiheelle. Devisen voi siis laskea olevan tärkein osittain siksi, että ilman hyvää suunnittelua ei toimi käyttöliittymä sen enempää kuin tutkiminenkaan, suunnitelmat täytyy olla selvät, jotta on järkevää lähteä jatkamaan sen pidemmälle. Kuitenkin kun keskittyminen on suuntautunut enemmän ketteriin malliehin, on muistettava, että tuotettaessa sykleissä myös testaus tapahtuu sykleissä. Ei ole mahdotonta muuttaa suunnitelmia myös sen mukaan, jos tuotantovaiheessa tapahtuu jokin radikaalisti aikatauluun tai implementointiin vaikuttava

muutos.

Devise on menetelmässä se osa, jota käytetään ensimmäisessä vaiheessa varsinkin jokaista testaus- tai tutkintajaksoa ennen. Sen avulla kartoitetaan suunta, johon halutaan koko käytettävyyttä suunniteltaessa lähteä, aina konventioista ja kuormittavuudesta lähtien. Se myös pitää sisällään koko projektin mallin ja siitä voidaan tarkastaa, kuinka alkuperäinen suunnitelma on toteutunut. Devise on vaihe, jossa käyttäjä kartoittaa sekä testauksen ja tutkimisen tarpeen.

Toisen vaiheen Devise kartoittaa viimeisen eli Improve- vaiheen jälkeen tehdyt mahdolliset muutokset ja seuraa suunnitelmaa toteutumisten osalta, joita taas Investigate-vaihe on kirjannut löytyneeksi. Käytännössä menetelmän alussa ja lopussa on Devise, hieman eri sisällöillä. Toisen, eli loppuvaiheen Devise- taas auttaa seuraavaa Investigatea keskittymään oikeaan suuntaan, mikäli muutosta suunnitelmaan on matkan varrella tehty. Ketterän menetelmän testauksessa ennakkosuunnitelmia ei välttämättä tehdä, kuten luvusta 2.3 on käynyt ilmi, joten Devise on käytännössä karkea punainen lanka siihen, mikä on suunta, kun aletaan tutkia.

6.3 Investigate

Investigate tarkoittaa tutkimista tai selvittämistä. Termi on otettu joukkoon juuri sen vahvuuden vuoksi. Tutkitaan kaikki mahdollisuudet virheiden ja ongelmien taholta; jottei yksikään kivi jäisi kääntämättä. Investigate- vaihetta ei ole mahdollista suorittaa, mikäli Devise on tekemättä. Investigate kuvaa menetelmässä testausta ja samalla myös sitä, kuinka paljon virheiden etsimisellä oikeasti voidaan saavuttaa ja miten paljon epäkohtia tuotteesta voikaan löytyä. On tärkeää, että käydään kaikki vaiheet läpi useaan otteeseen ja panostetaan siihen. Se, että sisällytetään menetelmän testaus osaksi projetin testausta ja jo suoraan scopeen, helpottaa ihmisiä ymmärtämään, miksi se kuuluu suorittaa ja mitä sillä tavoitetaan.

Investigate on menetelmässä se osa, jonka toiminta näkyy menetelmäkuvauksessa (liite

6.) keskeisimpänä, sillä se on selkeästi konkreettisesti toiminnallisin osuus menetelmän vaiheista. Sitä, että tutkiminen virheiden ja ongelmien varalta tapahtuu jokaisessa iteraatiossa uusien implementoitujen osioiden jälkeen, voisi pitää itsestään selvyytenä, onhan toiminnallinenkin testaus liitetty projekteihin tällä hetkellä iteraatiosidonnaisesti. Kuitenkin tärkeänä osana Investigaten käyttöä tulisi huomata, että kun aletaan olla saamassa tuotetta kokonaisuudeksi ja kasaan, on edessä vielä mielellään ainakin yksi, mahdollisesti useita Investigate- vaiheita. Näiden vaiheiden määrän määrää luonnollisesti projektipäällikkö, tarve ja aikataulu yhdessä. Aikatauluun kuitenkin on mallin mukaan täytynyt jättää tarpeen tullen aikaa Investigate- vaiheille, joiden määrää ei käytännössä voi ennustaa etukäteen. Jos jostakin syystä Investigate- vaiheille on jätetty liikaa aikaa, se tuskin koituu ongelmaksi, sillä ainahan voi parantaa jo olemassaolevaa tai kehittää ajan salliessa vielä lisää. Investigate on siis menetelmässä se osa, jonka avulla käyttäjä suorittaa testaamisen.

Investigate ei määritä varsinaisesti raja-arvoja tutkimiselle. Ainoa lähtökohta, jonka menetelmä asettaa on se, että tutkimuksen lähtökohtana on suoraan käyttötapaus itsessään. On muistettava, että usein tutkimus ja testaus luovat matkan varrella itse lisää testicaseja. Näin ollen on järkevää kohdentaa resurssit sillä hetkellä juuri ilmenevään ongelmaan tai virheeseen ja käyttää aika näiden ratkaisuun ja raportointiin. Ei ole vaarallista, jos jokaisella kerralla varattu aika ei täyty. Voi olla, että sen jääminen ”pankkiin” on hyväksi myöhempää kertaa varten. Investigate- vaiheessa menetelmän käyttäjä kirjaa ylös löytyneet ongelmat, kommunikoi ne muulle projektiryhmälle ja antaa mahdollisuuksien mukaan jopa kehitysehdotuksia Improve- vaihetta varten.

6.4 Improve

Kehittäminen on viimeinen, mutta ei missään nimessä mitätön osa DII- menetelmää, sillä mikäli suunnittelu ja tutkimus on hoidettu huolellisesti, voidaan tällä osa-alueella saada aikaiseksi suurimmat ja näkyvimmät vaikutukset ja muutokset käytettävyyteen ja sen parantamiseen. Mikäli tässä vaiheessa jätetään löytyneet virheet korjaamatta, on työ ollut turhaa ja resurssit käytetty tyhjään. Investigate- vaiheesta tulleiden mahdollisten

kehitysehdotusten suhteen käydään yhteinen keskustelu projektipäällikön kanssa asiasta, onko mahdollista toteuttaa ja millaisen ajan kyseinen parannus vaatii. Scopeen tulisi olla laskettu jokaisen iteraation kohdalle implementointiin parannusaika, jolloin Improvea voisi hyödyntää laadun parantamiseksi ennen seuraavaan kehitysvaihetta. Vaikka kommunikoinnin osuus ketterässä menetelmässä on merkittävä ja täysin aiheellinen, on kaikista pienistä Improve- vaiheen parannuksista turha kuormittaa koko projektiryhmää tai järjestää palaveria asiasta, vaan riittää keskustelu projektipäällikön kanssa asiasta. Muutosten ollessa kohtalaisia, jotka ovat tehtävissä Improve- vaiheelle varatun ajan puitteissa, on projekti kulkenut suunnitellusti oikeaa tietä ja näin ollen ylimääräinen informoiminen on tarpeetonta. Kun parannukset on tehty, kannattaa muutakin ryhmää kuitenkin informoida muutoksesta. Mikäli tarvittavat parannukset Improve- vaiheessa ovat kohtuuttoman suuria varattuun aikaan nähden, pyritään parannus jaottelemaan osiksi, jotka priorisoidaan ja toteuteaan tämän mukaan. Jäljelle jäävät voidaan hoitaa mahdolisuuksien mukaan muiden iteraatioiden mahdollisesta yliajasta, tai projektin loppupuolelle varatulla ajalla. Kuitenkin sellaiset muutokset, jotka vaikuttavat seuraavien iteraatioiden sisältöön ja tuotteen kehittämiseen kannattaa hoitaa pois mahdollisimman aikaisessa vaiheessa, jotta työ parannuksen suhteen on pienempi.

6.5 Esimerkki menetelmän käytöstä projektissa

Projektipäällikkö määrittelee tavoitteen projektille X ja tämän perusteella myöhemmin aikataulun, jonka puitteissa projektia X aletaan suunnittelemaan ja lopulta työstämään. Projektipäällikkö varaa aikataulusuunnitelmaa tehdessään jokaiseen iteraatioon, mielellään iteraation loppupuolelle tilan menetelmän hyödyntämiselle. Tämä tarkoittaa ajallisesti noin puolta tuntia per iteraatio. Projektin alussa käytetään Devise- vaiheen kanssa aikaa enemmän, kun karkea suunnitelma projektin kulusta ja tavoitteista on selvillä ja voidaan asettaa myös menetelmälle tavoite ja suunnitelma tarkemmin, niin koko projektin osalta, kuin yksittäisen iteraationkin osalta.

Käyttöliittymäsuunnittelija tekee ensimmäiset rautalankamockupit ja nämä ovat ensimmäiset Investigate- vaiheeseen menevät materiaalit. Palaute kerätään myös näistä rautalankamockupeista ja Improve- vaiheen aikana mallinnuksiin tehdään parannukset, jotta virheet eivät siirry eteenpäin tuotetta suunnitellessa. Arvioijana voi toimia tiimin toinen jäsen tai joku muu työtoveri. On järkevää kuitenkin huomioida, että varsinkin paperilla käytävässä tutkinta- ja testausvaiheessa täytyy jonkin verran olla tietoa tuotteen käyttötarkoituksesta jo etukäteen, sillä paperilla asiat eivät välttämättä aukea yhtä hyvin, kuin esimerkiksi näytettäessä arvioijalle prototyyppi koneelta linkityksien kera. Näin toimitaan mallinnusten kanssa siihen asti, kunnes jotakin jo implementoitua on. Tämän jälkeen arvioitavana on siis sekä mockuppeja, että HTML-pohjiakin ja myöhemmässä vaiheessa projektin edetessä valmista tuotetta. Investigate- vaiheen testaus- ja tutkintamateriaalin määrä ja taso nousee sitä mukaa, kun projekti etenee ja osioita tulee valmiiksi. Jokaisen kohdalla kuitenkin annetaan palaute virhe- ja ongelmatilanteista ja ne viedään yhtä tärkeinä Improve- vaiheeseen. Tällä tavalla jatkettaessa jokainen mockup, HTML-pohja ja valmiiksi tuotettu osio tuotteesta on suunniteltu, tutkittu ja parannettu. Käytännössä jokainen valmis osio on tutkittu kolmeen kertaan virhe- ja ongelmatilanteiden varalta ja vastaan tulleet virhe- ja ongelmatilanteet on parannettu mahdollisuuksien mukaan kolmeen kertaan eri tasolla.

Kun iteraatioittain on saatu lopulta testattua ja tutkittua kokonainen tuote, käydään se vielä näiden kolmen periaatteen kautta läpi kokonaisuutena. Tehdään uusi suunnitelma, uusi tutkimus ja vielä uudet parannukset. On toki mahdollista, ettei parannettavaa juuri ole, mutta ei siitä huolimatta kannata laskea sen varaan ja jättää vaihetta tekemättä. Tämän vaiheen vuoksi on tärkeää, että projektin loppuun, ennen loppukäyttäjätestausta on jätetty tilaa koko kolmivaiheiselle menetelmälle. Kun mahdolliset viimeisessä vaiheessa löytyneet virheet on korjattu, voidaan tuote päästää loppukäyttäjätestauksen ulottuville. Tässä vaiheessa tuotteen tulisi olla laadultaan ja käytettävyydeltään parempi ja selkeämpi, läpinäkyvämpi ja ongelmattomampi kuin ilman kaikkia kolmea läpikäytyä vaihetta. On mahdollista, mikäli projektin aikataulu sallii, että viimeinen kolmen vaiheen kokonaisuus tehdään toistamiseen tai useammin, jos näin halutaan ja aikataulut antavat sen periksi. Lopputuloksena on parannettu tuote ja parannettu käytettävyys.

Loppukäyttäjätestaukseen asti kolmivaiheisella menetelmällä voidaan tehdä parannuksia, mutta sen jälkeen toimitaan, kuten projekti normaalistikin, ilman DII-menetelmää toimisi saadessaan palautetta loppukäyttäjilta.

7 POHDINTA

Opinnäytetyöni on tehty tilaustyönä Polar Electro Oy:lle ja joka tuli luonnollisena jatkona työharjoittelulleni. Olen kiitollinen tästä mahdollisuudesta, joustamisesta työni aikana sekä heidän panoksestaan työhöni erityisesti ohjauksen ja kyselyyn vastanneiden suhteen. Opinnäytetyön tarkoituksena oli tutustua käytettävyyteen ja sen parantamiseen, osana ketterää ohjelmistokehitystä. Pohjana teoriaosuudessa käytiin läpi agile- menetelmien toimintamalleja yleisesti ja tutustuttiin erilaisiin käytettävyyden ohjenuoriin. Kokonaisuudessaan teoriapohjan tarkoitus on jalostaa niin tekijäänsä kuin lukijaakin ymmärtämään lopputuloksena syntynyttä menetelmää ja sen toimintamallia entistä paremmin, nimenomaan ketterien mallien ja ketterän testauksen pohjalta jokapäiväiseen käyttöön.

Kyselytutkimuksen vastausten saaminen ajoissa oli aluksi hieman haasteellista ja vaikutti omalta osaltaan jo muutenkin tiukan aikataulun kiireellisyyteen. Vastauksia otannasta tuli kuitenkin lopulta melko kiitettävästi ja ne olivat suhteellisen kattavia. Näiden perusteella lähtötilakartoituksesta saatiin tehtyä mielestäni kattava ja valmistava uuden menetelmän luomista varten juuri tähän tarkoitukseen, näille mahdollisille käyttäjille. Oli selkeästi huomattavissa alusta lähtien, kuinka haasteellinen aihe ja tehtävä kyseessä on, mutta tyytyväisyyteni lopputulokseen on suurempi kuin koskaan oletin. Toivottavasti se johtuu ennemminkin onnistumisesta, kuin pelkästä itsensä likoon asettamisesta.

Tutkimuksesta selvisi, ettei tällä hetkellä ole olemassa keinoa ohjelmistotuotannon puolella, jolla käytettävyyttä kartoitettaisiin tai mitattaisiin ja tämä antoikin hyvät mahdollisuudet innovaatioille matkan varrella, joista lopulta kasvoi kehittämäni menetelmä, DII. Käytännössä toivon joku päivä käyttäväni työni ohessa Devise, Investigate ja Improve menetelmää, sillä koen UI-suunnittelijana olevani vasta lapsen kengissä ja silti ymmärtäväni käytettävyyden tärkeyden. DII- menetelmälle näen käyttöä niiden kokemusten puitteissa, joita minulla ketterien menetelmien

ohjelmistokehitysprojekteista on. Se ei ole paljon, mutta menetelmää voisi pitää suuntaa antavana ja varmasti opettavana työkaluna jo pelkästään suunnittelun näkökulmastakin. Olisi hienoa nähdä, kuinka menetelmä toimisi oikeasti käytännössä, omana osanaan projektia suunnitellusti ja organisoidusti.

Prosessina koko opinnäytetyö on ollut äärimmäisen kasvattava, työn yhdistäminen vaativaan prosessiin on ollut kuluttava, kasvattava ja ennenkaikkea siitä on ollut vaikea luopua. Toisaalta olisikin mielenkiintoista nähdä, millaisia tutkimuksia tästä menetelmästä käytännössä voitaisiin saada tai voisiko menetelmän käyttöönottoa seurata jotenkin. Voisikohan tässä olla polku, jota joku muu voisi kulkea jälkeeni vaikkapa opinnäytetyönsä merkeissä?

LIITTEET

Liite 1. Kysely

Kyselyn pohja, joka lähetettiin valikoidulle joukolle Polar Electro Oy:n sisällä

Jyväskylän Ammattikorkeakoulun opintosuunnitelmaan kuuluva Opinnäytetyö, tekijänä Heli Puikkonen

Kaikkia vastauksia käsitellään osana opinnäytetyötä ”Käytettävyystestaus osana ketterää ohjelmistokehitystä” ja ne ovat tärkeänä osana tutkimusta. Vastaajat ovat työssä anonyymejä otannassa. Kiitos panoksestasi!

Kyselyn kaikki kysymykset pohjautuvat **ohjelmistotuotannon** asettamaan näkökulmaan käytettävyydestä, käytettävyystestauksesta ja sen tämän hetkisestä tilasta. Toivon, että vastaajat huomioivat tämän näkökulman vastatessaan.

1. Miten käytettävyyttä mitataan tällä hetkellä?
2. Miten käytettävyystestaus hoidetaan tällä hetkellä?
3. Kuinka saatuja tuloksia tutkitaan ja tulkitaan? Vai tutkitaanko ollenkaan?
4. Onko huomattavissa eroa vesiputousmallilla ja ketterällä menetelmällä tehdyn projektin käytettävyyden ja käytettävyystestauksen välillä?
5. Millaisia ongelmakohtia käytettävyyteen ja käytettävyystestaukseen liittyen olet huomannut projektien aikana?
6. Kuinka siirtymä vesiputousmallista ketterään on tiimissä otettu vastaan? Onko vaikuttanut työskentelyyn tai tiimidynamiikkaan?
7. Kehitysideoita?

Liite 2. Kahdeksan kultaista sääntöä, Shneidermann, Ben

1. **Yhteneväisyys:**
   Toimintaketjun ja toimintamallien yhteneväisyys niin fonttien, värien kuin ikonien ja kuvienkin kesken.
2. **Oikotiet:**
   Edistyneille käyttäjille tarjotaan makroja ja näppäinoikoteitä.
3. **Palaute:**
   Informatiivisen palautteen antaminen käyttäjälle.
4. **Lopputulos:**
   Dialogien suunnittelussa otettava huomioon, että niiltä löytyvät alku, keskikohta ja loppu. Lopputulos täytyy löytyä.
5. **Virheenkäsittely:**
   Virheiden estäminen ja yksinkertainen virheenkäsittely helpottavat käyttäjää toipumaan virhetilanteista.
6. **Peruminen:**
   Käyttäjän täytyy kyetä perumaan toiminta tai toimintasarja helposti.
7. **Kontrolli:**
   Käyttäjän täytyy voida tehdä valintoja ja keskeytyksiä, sekä käynnistää tapahtumia ja toimintoja.
8. **Kuormitus:**
   Käyttäjän muistin kuormittamista täytyy välttää ja jakaa vain tarvittava informaatio näytöllä.

Liite 3. Ten Usability Heuristics, Nielsen, Jakob

These are ten general principles for user interface design. They are called "heuristics" because they are more in the nature of rules of thumb than specific usability guidelines.

1. **Visibility of system status**

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

2. **Match between system and the real world**

The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

3. **User control and freedom**

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

4. **Consistency and standards**

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

5. **Error prevention**

Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

6. **Recognition rather than recall**

Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

7. **Flexibility and efficiency of use**

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

8. **Aesthetic and minimalist design**

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

9. **Help users recognize, diagnose, and recover from errors**

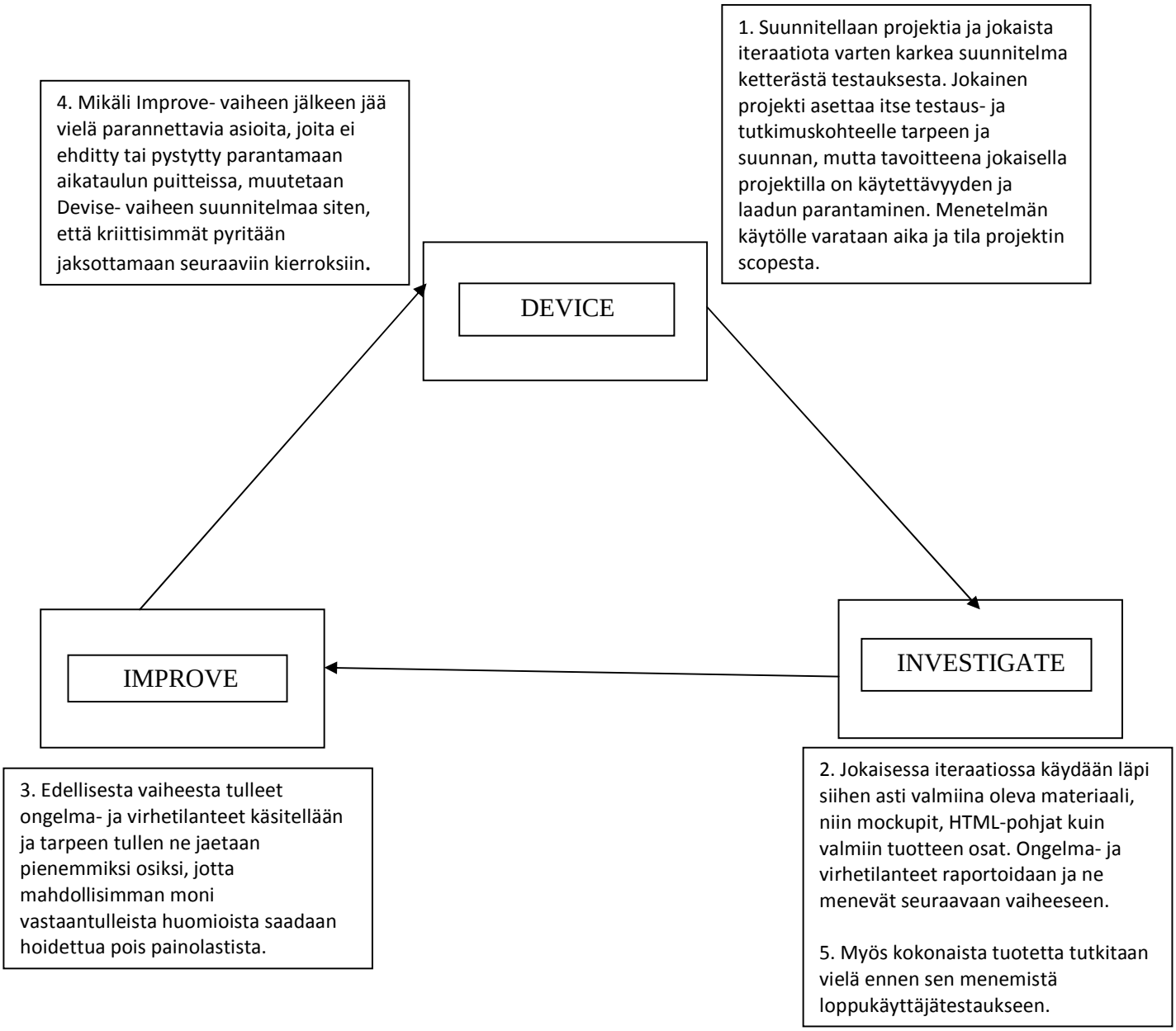
Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

10. Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

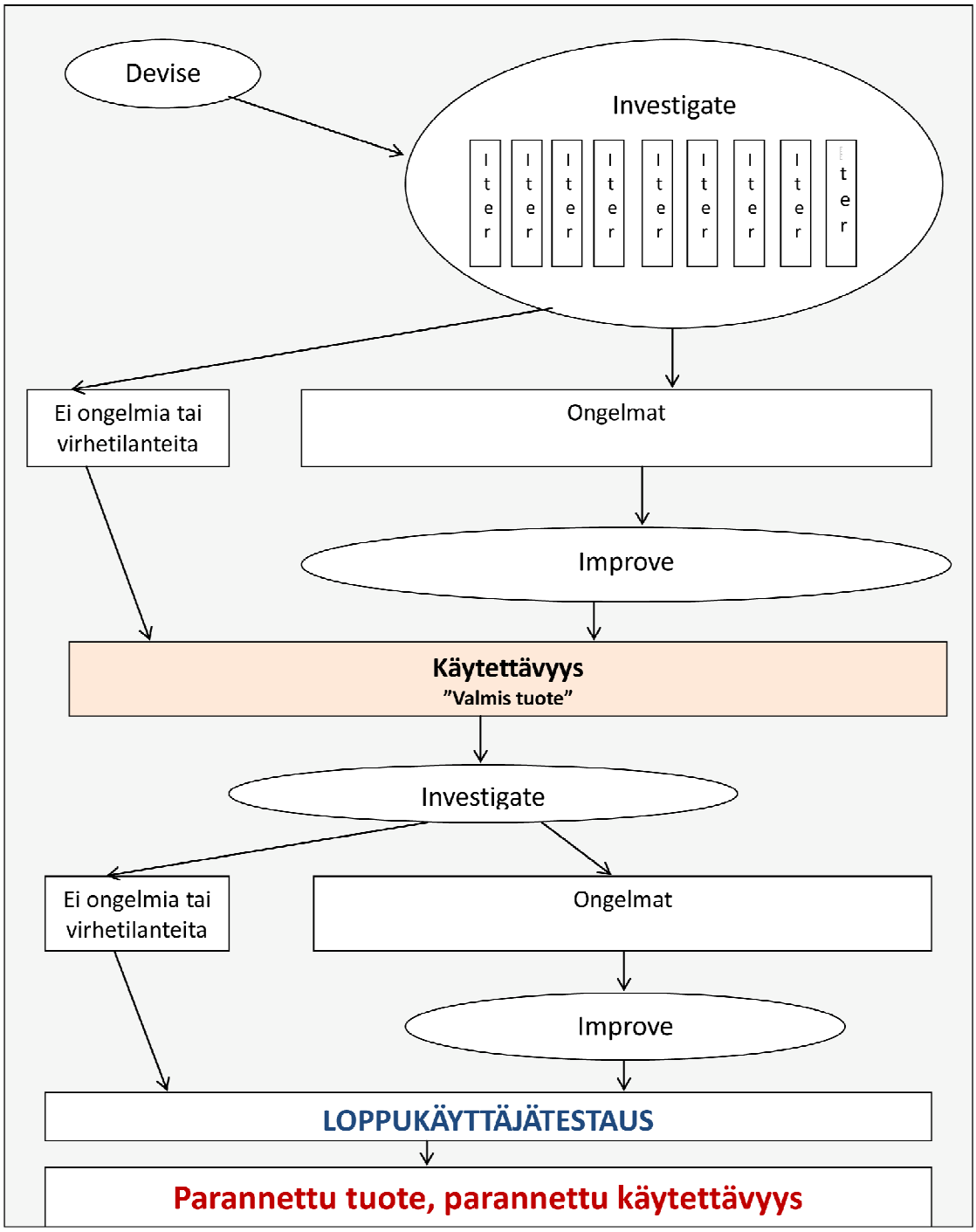
Liite 4. Agile malli

Liite 5. DII-menetelmän kaavio



Liite 6. Menetelmäkuvaus

Harmaa alue kuvaa koko projektia.



LÄHTEET

Hirsjärvi, S., Remes, P. & Sajavaara, P. 2006. Tutki ja kirjoita. Helsinki: Tammi

Johdatus tietojärjestelmiin. Viitattu 4.10.2011.
http://www.okol.org/verkkokurssit/datanomi/tietojarjestelmien_kaytto_ja_kehittamine n/johdatus_tietojarjestelmiin/kehittamistyon_vaiheet_ja_elikaarimallit/kehittamistyon_ vaiheet_ja_elinkaarimallit_asia.htm

Ketterä ohjelmistokehitys. Viitattu 4.10.2011
https://koppa.jyu.fi/avoimet/thk/muut/agile-ja-trac/agile

Ketterä testaus ja testaus ketterässä ohjelmistokehityksessä, 6.10.2010.
Viitattu 9.11.2011
http://mattivuori.net/julkaisuluettelo/liitteet/kettera_testaus.pdf

Koskinen, J. 2005. Käytettävyystestaus. Tampereen yliopisto, Tietojenkäsittelyn laitos

Kosonen, K. 2005. Käytettävyystutkimuksen menetelmien vertailu. Tampereen yliopisto, Tietojekäsittelyn laitos

Krug, S. 2006. Don't make me think! USA: New Riders Publishing

Krug, S. 2010. Rocket surgery made easy. USA: New riders publishing

Käytettävyys, kuvien käyttö. Viitattu 9.11.2011.
http://ww.cs.tut-fi/~sevi/luennot/seviluennot4.pdf

Käytettävyys käyttöliittymäsuunnittelussa. Viitattu 4.11.2011
http://www.cs.tut.fi/~grako/2008/luennot/grako_kayt_luento.pdf

Nielsen, J. 1993. Usability engineering. New York: Academic press.

Nielsen, J. 2000 WWW-suunnittelu. Jyväskylä: Gummerus

Nielsen, J. 2005. Ten usability heuristics. Viitattu 29.8.2011.
http://www.useit.com/papers/heuristic/heuristic_list.html

Plan, Do, Check, Act. Viitattu 13.11.2011.
http://laatu.tkk.fi/fi/toimintakasikirja/pdca-plan-do-check-act.pdf 13.11.11.)

Suunnittelu, toteutus, testaus, hyväksyntä. Viitattu 13.11.2011
http://batman.jamk.fi/~tuito/www_sk/spiraali.gif